386 | ASM
Reference Card

© 1986-1991 Phar Lap Software, Inc.

Phar Lap Software, Inc.
60 Aberdeen Ave., Cambridge, MA 02138
(617) 661-1510
FAX (617) 876-2972
dox@pharlap.com
tech-support@pharlap.com

Conventions

courier	indicates command line and switch syntax
italics	indicate a name or value that must be entered by the user
[ ]	(brackets) indicate optional characters in switches
{ }	(braces) indicate optional items in directives
, . . .	indicates an item or items that may be repeated
|	(vertical bar) separates alternative items

Assembler Directives

.186
   Enables assembly of 80186 instructions.

.286
   Enables assembly of 80286 nonprotected instructions.

.286p
   Enables assembly of all 80286 instructions.

.287
   Enables assembly of 80287 instructions.

.386
   Enables assembly of 80386 nonprotected instructions. This is the default mode.

.386p
   Enables assembly of all 80386 instructions.

.386r
   Enables assembly of 80386 instructions for real mode.

.387
   Enables assembly of 80387 instructions.

.486
   Enables asembly of 80486 nonprotected instructions.

.486p
   Enables assembly of all 80486 instructions.

.486r
   Enables assembly of 80486 instructions for real mode

.8086
   Enables assembly of 8086 instructions.

.8087
   Enables assembly of 8087 instructions.

name = *expression*
 Creates the symbol *name* with the value of *expression*. The symbol will be
 a constant, variable, or label.

ALIGN *value*
 Aligns the location counter on a power of two boundary.

ASSUME *registername:segname*
 Indicates to the assembler that *registername* will point at the segment or
 group named *segname* during the execution of the program. If *segname*
 is NOTHING, the register is assumed to have an undefined value.

COMMENT *delim text delim*
 Causes the assembler to treat all *text* between the two occurrences of
 delim as a comment. *Text* may include carriage returns, so multi-line
 comments are possible.

COMM {NEAR|FAR} *name:size*{:*count*}
 Allocate communal variable.

{*name*} DB *value,...*
 Allocates and initializes one byte of memory for each *value*. If the
 optional name is present, a variable of type BYTE is created with the
 specified name.

{*name*} DD *value,...*
 Allocates and initializes a double word (four bytes) of memory for each
 value. If the optional *name* is present, a variable of type DWORD is
 created with the specified name.

{*name*} DF *value,...*
 Allocates and initializes three words (six bytes) of memory for each *value*.
 If the optional *name* is present, a variable of type PWORD is created with
 the specified name.

{*name*} DP *value,...*
 Synonym for {*name*} DF *value,...*

{*name*} DQ *value,...*
 Allocates and initializes a quad–word (eight bytes) of memory for each
 value. If the optional *name* is present, a variable of type QWORD is
 created with the specified name.

{*name*} DT *value,...*
 Allocates and initializes ten bytes of memory for each *value*. If the
 optional *name* is present, a variable of type TBYTE is created with the
 specified name.

{*name*} DW *value,...*
 Allocates and initializes one word (two bytes) of memory for each *value*.
 If the optional *name* is present, a variable of type WORD is created with
 the specified name.

ELSE
 Indicates the end of the true portion and the start of the false portion of a
 conditional block.

END {*entry*}
 Indicates the end of the program module being assembled and optionally
 establishes the program's entry point as *entry*.

ENDIF
 Signals the end of a conditional block.

ENDM
 Signals the end of a macro or repeat block.

name ENDP
 Signals the end of the procedure definition for *name*.

name ENDS
 Closes the segment named *name* or ends a structure definition for *name*.

name EQU *expression*
 Assigns the value of *expression* to *name*. *Expression* can be an address
 expression, an assembler keyword, an arbitrary string of text, or a
 constant value.

EVEN

Aligns the location counter for the currently open segment to an even value by generating a NOP instruction (90H), if necessary.

EXITM

Forces the current macro expansion or repeat block to terminate immediately.

EXTRN *name:type,...*

Signals to the assembler that the symbol named *name* is external to the current module and is of type *type*.

name **GROUP** *segname,...*

Creates a definition for a group name with members from the list of one or more segments following the directive.

IF *expression*

Assembles instructions in the true portion of a conditional block if *expression* is true (evaluates to a non-zero value).

IFB *<string>*

Assembles instructions in the true portion of a conditional block if *string* is blank.

IFDEF *name*

Assembles instructions in the true portion of a conditional block if there is a symbol named *name* in the assembler's symbol table, and it was defined before the current source line.

IFDIF *<str1>, <str2>*

Assembles instructions in the true portion of a conditional block if *str1* and *str2* are different.

IFE *expression*

Assembles instructions in the true portion of a conditional block if *expression* is false (evaluates to 0).

IFIDN *<str1>, <str2>*

Assembles instructions in the true portion of a conditional block if *str1* and *str2* are identical.

IFNB *<string>*

Assembles instructions in the true portion of a conditional block if *string* is not blank.

IFNDEF *name*

Assembles instructions in the true portion of a conditional block if there is no symbol named *name* in the assembler's symbol table, or if *name* was defined after the current source line.

INCLUDE *filename*

Inserts the text contained in the source file specified by *filename* into the file currently being assembled.

IRP *fparam, <aparam,...>*

Start of repeat block that will be repeated once for each actual parameter *aparam*, with the actual parameter value being substituted for the formal parameter *fparam* in the text of the repeat block.

IRPC *fparam, string*

Start of repeat block that will be repeated once for each character in *string*, with the current character being substituted for the formal parameter *fparam* in the text of the repeat block.

name **LABEL** *type*

Creates a new variable or instruction label of type *type* with a value equal to the location counter for the current segment.

.LALL

Causes all statements in macro expansions to be listed in the assembler listing file.

.LFCOND

Causes text in false conditional blocks to be copied into the assembler listing file even though it is not being assembled.

Assembler Directives (cont.)

`.LIST`
Enables listing of program statements in the assembler listing file.

`.LISTI`
Enables listing of source lines from files included with the INCLUDE directive.

`LOCAL dname,...`
Creates one or more dummy names for use within a macro. The dummy name is replaced by an assembler-generated unique name of the form ??XXXX, where X is a hexadecimal digit when the macro is expanded.

`name MACRO fparam,...`
Start of a macro definition block for macro *name,* with formal parameter names *fparam.*

`NAME modulename`
Sets the name of the module being assembled to the first 132 characters of *modulename.*

`.NOSIGNLOGI`
Disable generation of undocumented sign-extended form of arithmetic instructions.

`ORG expression`
Sets the value of the location counter for the currently open segment to *expression.*

`%OUT text`
Writes text to the user's terminal on both pass one and pass two of the assembly.

`%OUT1 text`
Writes *text* to the user's terminal on pass one of the assembly.

`%OUT2 text`
Writes *text* to the user's terminal on pass two of the assembly.

`PAGE {length}, {width}`
Sets the page length and width for the listing file to *length* and *width,* respectively.

`PAGE +`
Increments the section number used on the header line of listing file pages and generates a page break.

`PAGE`
Generates a page break in the assembler listing file.

`name PROC type`
Initiates definition of a procedure called *name* of type *type.*

`.PROT`
Enables assembly of protected instructions for the 80386 or 80286.

`PUBLIC name...`
Makes the list of variables, instruction labels, or absolute symbols available to all other modules in the program.

`.PUBPARSE char`
Remove specified leading character from public symbol definition records in object module.

`PURGE macroname`
Has no effect.

`.RADIX expression`
Sets the default radix for numbers in the input file to *expression.* Numbers in *expression* are always evaluated in base 10, regardless of the current default radix.

`name RECORD fname:width {=expr},...`
Defines a record called *name* with bit fields named *fname.* The field width, in bits, is given by the constant value *width* and the *=expr,* if present, defines the default value for the field.

Assembler Directives (cont.)

REPT *expression*
    Beginning of a block to be repeated *expression* number of times.

.SALL
    Suppresses the listing of macro expansions in the assembler listing file.

name SEGMENT {*align*} {*combine*} {*useatr*} {*access*} {*'class'*}
    Opens a program segment called *name* with attributes *align, combine, useatr, access,* and *class.*

.SIGNLOGI
    Enable generation of undocumented sign-extended form of arithmetic instructions.

.SFCOND
    Suppresses the listing of subsequent false conditional blocks in the assembler listing file.

name STRUC
    Signals the beginning of a structure definition.

SUBTTL {*text*}
    Sets the subtitle to be used on page headers in the listing file to *text.*

.TFCOND
    Toggles the current state of the "list false conditional blocks" flag.

TITLE *text*
    Sets the title to be used on page headers in the listing file to *text.*

.XALL
    Causes only statements in macro expansions which generate object code to be listed in the assembler listing file.

.XLIST
    Suppresses listing of subsequent source lines in the assembler listing file.

.XLISTI
    Disables listing of source lines from files included with the INCLUDE directive.

Operator Precedence

Operator	Precedence
LENGTH, SIZE, WIDTH, MASK, [ ]	highest
. (structure field name)	
: (segment override)	
OFFSET, SEG, TYPE, THIS	
HIGH, LOW, HIGHW, LOWW	
unary +, unary −	
PTR	
*, /, MOD, SHL, SHR	
+, −	
EQ, NE, LT, LE, GT, GE	
NOT	
AND	
OR, XOR	
SHORT, .TYPE	lowest

Command Line Switches

386ASM

-CV[SYM]	Generate Code View symbol records in object file.
-D[EFINE] *name{=string}*	Define a text symbol.
-E[RROR]L[IST] *file*	Object file name.
file1,file2 . . . ,file*n*	Input file names.
-FULLWARN	Enable extra error checking.
-I[NCLUDE] *path*	Set INCLUDE directive path.
-L[IST] *file*	Listing file name.
-LNUM	Generate line number records in object file.
-NOD[ELETE]	Produce object file even if errors.
-NOL[IST]	Suppress listing file.
-NOO[BJECT]	Suppress object file.
-NOSIGNLOGI	Disable sign-extended arithmetic instructions.
-NOS[YM]	Suppress symbol table in listing.
-NO87	Select no numeric coprocessor.
-[80]87	Select 8087 numeric coprocessor.
-[80]287	Select 80287 numeric coprocessor.
-[80]387	Select 80387 numeric coprocessor.
-O[BJECT] *file*	Object file name.
-ONEC[ASE] -TWOC[ASE]	Set case of symbols.
-PUBPARSE *char*	Remove leading character from public symbols.
-SIGNLOGI	Enable sign-extended arithmetic instructions.
-[80]86	Select 8086 CPU.
-[80]186	Select 80186 CPU.
-[80]286	Select 80286 CPU.
-[80]386	Select 80386 CPU.
-[80]486	Select 80486 CPU.
-[80]286P	Select 80286 CPU with privileged instructions.
-[80]386P	Select 80286 CPU with privileged instructions.
-[80]486P	Select 80286 CPU with privileged instructions.
-[80]386R	Assemble 386 instructions for real mode.
-[80]486R	Assemble 486 instructions for real mode.